Un modèle neuronal pour la régression et la discrimination sur données fonctionnelles

Fabrice Rossi, Brieuc Conan-Guez

Un modèle neuronal pour la régression et la discrimination sur données fonctionnelles[0]

Fabrice Rossi[*†] **& Brieuc Conan-Guez**[†]

∗ *CEREMADE, UMR CNRS 7534,*

Université Paris-IX Dauphine,

Place du Maréchal de Lattre de Tassigny,

75016 Paris, France

Fabrice.Rossi@dauphine.fr

† *Projet AxIS,*

INRIA-Rocquencourt,

Domaine de Voluceau, BP 105 Bâtiment 18

78153 Le Chesnay Cedex, France

Brieuc.Conan-Guez@inria.fr

RÉSUMÉ. *L'objectif de cet article est de présenter un nouveau modèle neuronal adapté à la régression sur variables explicatives fonctionnelles. Nous décrivons le perceptron multi-couches (PMC) fonctionnel et résumons les propriétés théoriques de celui-ci, en terme de puissance de calcul et d'estimation des paramètres. Nous proposons ensuite deux techniques de mise en œuvre informatique que nous illustrons dans deux applications, l'une sur des données artificielles, l'autre en spectrométrie. Le PMC fonctionnel est ainsi comparé au PMC classique ainsi qu'à des méthodes plus classiques adaptées elles aussi aux données fonctionnelles. Dans les exemples choisis, les PMC fonctionnels obtiennent de meilleures performances que les autres techniques et représentent la fonction de régression de façon très parcimonieuse.*

0. Paru dans la Revue de Statistique Appliquée, Vol. LIII No. 4, pages 5 à 30, 2005.

ABSTRACT. *We introduce in the present article a new neural model adapted to regression problems in which explanatory variables are functions. The paper describes the functional Multi-Layer Peceptron (MLP) and summarizes its main theoretical properties: universal approximation and consistent parameter estimation. We propose two possible numerical implementations of the model and show how to apply them to artificial data (Breiman waves) and to a spectrometric problem. These application examples show that functional MLPs achieve better performances than other techniques while using a very low number of numerical parameters.*

MOTS-CLÉS : *Analyse de données fonctionnelles, Réseaux de neurones, Perceptrons Multi-Couches, Régression, Discrimination, Approximation universelle, Consistance, Spectrométrie.*

KEYWORDS: *Functional Data Analysis, Neural Networks, Multi-Layer Perceptrons, Regression, Discrimination, Universal approximation, Consistency, Spectrometry.*

1. Introduction

Dans de nombreuses situations, il est naturel de décrire un individu par une ou plusieurs fonctions. Le cas le plus classique est celui d'un individu observé pendant un certain temps au cours duquel plusieurs mesures sont effectuées. On peut considérer par exemple plusieurs indices boursiers étudiés pendant une même période : chaque place boursière est observée par l'intermédiaire d'une fonction qui à une date associe la valeur de clôture des indices de cette place. Un autre exemple naturel est celui des mesures climatiques. Comme dans l'exemple boursier, une région peut être représentée par une fonction qui à une date associe des grandeurs comme la température moyenne de la journée correspondante, la quantité de précipitation, etc. Quand on dispose de données à haute résolution, on peut aussi prendre un point de vue géographique, la région étant alors décrite par une fonction qui à des coordonnées géographiques associe les grandeurs climatiques observées. On peut ainsi étudier la série temporelle des fonctions quotidiennes. Nous présenterons dans le présent article un autre exemple d'application concrète, l'étude du spectre d'absorbance d'échantillons de viande. Chaque individu est alors décrit par le spectre d'absorbance obtenu par un instrument travaillant sur 100 fréquences dans le proche infra-rouge.

[RAM 97] présente comment les méthodes classiques de l'analyse de données (Analyse en composantes principales, régression linéaire, etc.) ont été adaptées au cas des données fonctionnelles. Parmi les méthodes non citées par cet ouvrage car plus récentes, on peut évoquer entres autres l'extension des approches de type nuées dynamiques ([ABR 03]), la régression inverse par tranches ([FER 03b, DAU 01]) et les approches non-paramétriques basées sur des estimateurs à noyaux (utilisées en discrimination/régression, e.g., [FER 01], [FER 02b] et [FER 03a] ou pour une modélisation auto-régressive, e.g., [BES 00]). On pourra aussi se reporter à [BES 03] pour une vision générale récente. Nous nous intéressons ici à un cas particulier de modélisation de données fonctionnelles, celui dans lequel nous avons une variable à expliquer. Plus précisément, nous supposons que les individus décrits par une ou plusieurs variables fonctionnelles sont associés à une grandeur à prédire qui peut être une appartenance à une classe ou une valeur numérique au sens large (les vecteurs de $\mathbb{R}^n$ peuvent être traités). Il s'agit donc soit de pratiquer de la discrimination de fonctions (ou de n-uplets de fonctions), soit de faire de la régression multiple sur variables explicatives fonctionnelles. Nous excluons ainsi de notre champ le problème de la classification (cf [ABR 03]).

Dans ce cadre, nous proposons dans le présent article un nouvel outil pour le traitement des données fonctionnelles : le perceptron multi-couches (PMC) fonctionnel. Cet outil est une adaptation au cadre fonctionnel des perceptrons multi-couches définis pour les données numériques classiques. Les avantages des PMC classiques sont connus : ils sont capables d'approcher arbitrairement bien n'importe quelle fonction régulière (cf par exemple [LES 93], [HOR 93] et [STI 99]) de façon parcimonieuse ([BAR 93]). En pratique, les PMC donnent de meilleurs résultats que les méthodes linéaires (cf par exemple [ZHU 97, BAR 97]). De plus les techniques d'estimation des paramètres des PMC sont statistiquement valides (cf [WHI 89] pour une preuve de consistance des estimateurs et [WHI 90] pour une utilisation non paramétrique).

Nous verrons dans la section 2 que le passage à l'aspect fonctionnel est obtenu assez simplement par une généralisation du calcul réalisé par les neurones de la première couche du réseau. La section 3 sera consacrée à l'implémentation réelle du modèle proposé, au moyen de deux solutions. La première technique proposée est une approche directe assez souple et originale dans le cadre de

l'analyse de données fonctionnelles (ADF), permettant l'utilisation de représentations non linéaires de certaines fonctions manipulées. La deuxième technique, plus classique en ADF consiste à utiliser une projection sur une base de splines (ou plus généralement sur une base topologique de l'espace fonctionnel considéré) pour manipuler les fonctions. Nous verrons aussi dans cette section comment diverses techniques utilisées pour la recherche d'un modèle optimal basé sur un PMC classique peuvent s'appliquer au cas fonctionnel. Nous conclurons cette section par une brève évocation des propriétés théoriques des PMC fonctionnels, en particulier la propriété d'approximation universelle et la possibilité de construire des estimateurs consistants des paramètres d'un PMC fonctionnel.

Nous conclurons cet article par une application des PMC fonctionnels à des données simulées et à des données spectrométriques, présentée dans la section 4. Nous verrons que les PMC fonctionnels semblent très parcimonieux comparés aux approches non-paramétriques concurrentes, tout en offrant une qualité de modélisation comparable.

2. Le perceptron multi-couches fonctionnel

2.1. *Un PMC numérique*

Rappelons tout d'abord la définition d'un perceptron multi-couches numérique (nous renvoyons le lecteur entres autres à [ZHU 97, BAR 97, BIS 95] pour une présentation plus détaillée). Il s'agit d'une famille de modèles paramétriques obtenue par combinaison de modèles élémentaires non linéaires. Plus précisément, on définit un neurone numérique à n entrées comme un modèle paramétrique à $n+1$ paramètres réels représenté par la fonction de $\mathbb{R}^n$ dans $\mathbb{R}$ définie par : $N(x) = T(w.x+b)$, où T est la fonction d'activation (une fonction de $\mathbb{R}$ dans $\mathbb{R}$), w le vecteur de poids synaptiques ($w \in \mathbb{R}^n$) et b le seuil (un réel), le vecteur (w, b) constituant les paramètres du modèle. La fonction T est choisie non linéaire et relativement régulière afin d'assurer la propriété d'approximation universelle pour le PMC complet (cf [HOR 93] pour des conditions minimales sur T).

Le PMC est obtenu en combinant les neurones. La première étape est la combinaison en couches qui consiste à utiliser p neurones à n entrées en parallèle afin de construire un modèle paramétrique à

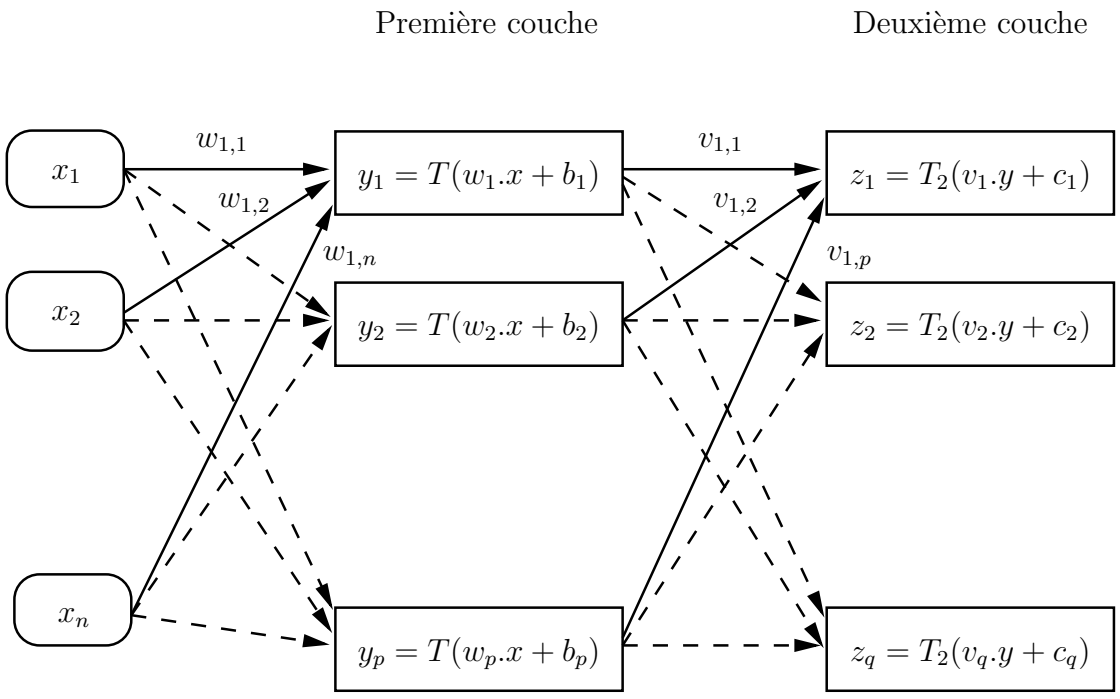

FIG. 1 – Un perceptron à deux couches

$(n+1)p$ paramètres qui transforme un vecteur de $\mathbb{R}^n$ en un vecteur de $\mathbb{R}^p$ selon l'équation $H(x) = y$ avec $y_i = T(w_i.x + b_i)$, où (w_i, b_i) désigne les paramètres du neurone i $(1 \leq i \leq p)$. La deuxième étape consiste à combiner les couches en composant les fonctions réalisées par celles-ci. Si on ajoute par exemple une deuxième couche à q neurones, on obtient globalement un modèle paramétrique à $(n+1)p + (p+1)q$ paramètres représenté par la fonction de $\mathbb{R}^n$ dans $\mathbb{R}^q$ définie par $G(x) = z$ avec $z_j = T_2(v_j.y + c_j)$ et $y_i = T(w_i.x + b_i)$. Dans cette expression, T_2 désigne la fonction d'activation de la deuxième couche, alors que les $(v_j, c_j)_{1 \leq j \leq q}$ sont les paramètres des neurones de cette couche (cf la figure 1 pour une représentation graphique du modèle). On règle la puissance du PMC (c'est-à-dire ses capacités d'approximation) grâce au nombre de couches et au nombre de neurones par couche. Le nombre d'entrées de la première couche correspond au nombre de variables explicatives (à valeurs réelles), alors que le nombre de sorties de la dernière couche correspond au nombre de variables à expliquer (toujours à valeurs réelles). Le nombre de couches et le nombre de neurones par couche constituent l'architecture du PMC.

Bien qu'un PMC d'architecture fixée soit un modèle paramétrique, il est possible de faire une utilisation non paramétrique d'une classe de PMC. En effet, les observations ne contraignent qu'une partie de l'architecture du PMC (le nombre d'entrées de la première couche et le nombre de neurones de la dernière couche) : une infinité d'architectures sont compatibles avec ces contraintes. Même si on se limite à deux couches, le nombre de neurones de la première reste totalement libre et peut être interprété comme une mesure de la complexité des données. Pour fixer les idées, considérons une suite d'observations dans $\mathbb{R}^n$, les u_i, réalisations des variables aléatoires U_i, associées à une variable à prédire, les s_i (des éléments de $\mathbb{R}$), réalisations des variables S_i. Le modèle de régression associé à un PMC à deux couches comportant p neurones dans la première couche est alors :

$$s_i = \sum_{k=1}^{p} v_k T(w_k.u_i + b_k) + c + \epsilon_i, \tag{1}$$

où ϵ_i désigne un bruit centré. Dans ce modèle, nous avons choisi pour T_2 la fonction identité. Les paramètres de ce modèle sont les v_k (des réels), c (un réel), les b_k (des réels) et enfin les w_k (des vecteurs de $\mathbb{R}^n$). Pour p fixé, nous avons donc un modèle paramétrique non linéaire. En pratique, comme pour tous les modèles paramétriques, le choix du modèle pose problème car il induit des hypothèses fortes sur les données. L'intérêt des PMC est qu'en faisant varier l'architecture (ici p, le nombre de neurones de la première couche), on explore une classe de fonctions très large (en fait toutes les fonctions régulières, cf par exemple [LES 93], [HOR 93] et [STI 99]). En liant le nombre de neurones de la première couche au nombre de données observées (comme dans [WHI 90]), on obtient une version non paramétrique des PMC dont on montre qu'elle permet une estimation consistante de l'espérance conditionnelle des variables à expliquer sachant les variables explicatives. Plus précisément, si les U_i sont identiquement distribuées et si les S_i sont aussi identiquement distribuées, White montre dans [WHI 90] qu'on peut construire une suite de fonctions, calculées par des PMC qui converge vers $E(S_0|U_0)$.

En pratique, comme nous le rappellerons plus en détail dans la section 3.5, on procède à une sélection de modèles qui consiste à choisir l'architecture du PMC en estimant ses performances par une technique de type validation croisée, ce qui n'a d'ailleurs rien de spécifique aux réseaux de neurones.

2.2. *Un PMC fonctionnel*

L'extension du neurone numérique au cas fonctionnel ne pose aucun problème, en remplaçant le produit scalaire $w.x$ dans $\mathbb{R}^n$ par son équivalent dans l'espace fonctionnel considéré, ce qui a été proposé dans [SAN 96a], [SAN 96b] et [STI 99]. Plus précisément, étant donnée une mesure σ-finie μ définie sur un espace mesurable X, un neurone de $L^p(\mu)$ dans $\mathbb{R}$ calcule une fonction N donnée par :

$$N(g) = T\left(b + \int fg \, \mathrm{d}\mu\right), \tag{2}$$

où T et b ont les mêmes définitions que pour un neurone numérique, et où f est une **fonction de poids** (qui remplace donc le vecteur de poids du neurone numérique). Si $f \in L^q(\mu)$, où q est l'exposant conjugué de p, $N(g)$ est définie pour tout $g \in L^p(\mu)$. De façon plus générale, on peut considérer une partie de $L^p(\mu)$ (ou un autre espace fonctionnel défini sur X) et donc un ensemble plus général de fonctions de poids acceptables. De même, il est parfaitement possible d'étendre le schéma proposé pour définir un neurone à plusieurs entrées fonctionnelles (cf [CON 02, ROS 05]) ou à des entrées mixtes.

Comme le neurone fonctionnel proposé ci-dessus produit une sortie réelle, la construction d'un PMC fonctionnel (PMCF) ne pose pas de problème : il suffit de commencer par une première couche constituée de neurones fonctionnels, puis d'utiliser exclusivement des neurones numériques dans les couches suivantes. Le cas le plus simple est celui d'un perceptron à deux couches (la première contient ici p neurones et la deuxième un seul), avec une entrée fonctionnelle et une sortie numérique. Un tel réseau calcule la fonction suivante :

$$H(\theta, g) = \sum_{k=1}^{p} a_k T\left(b_k + \int f_k g \, \mathrm{d}\mu\right) + c, \tag{3}$$

où θ est un vecteur qui regroupe tous les paramètres du PMCF, à la fois numériques (les a_k, les b_k et c) et fonctionnels (les f_k). Ceci correspond au modèle de régression suivant :

$$s^i = H(\theta, g^i) + \epsilon^i, \tag{4}$$

où les g^i sont des observations fonctionnelles (la variable explicative fonctionnelle), les s^i des observations réelles (la variable réelle à expliquer) et les ϵ^i des réalisations d'un bruit centré. On obtient ainsi une version fonctionnelle du modèle de régression utilisé dans le cas classique (équation 1).

Comme nous l'avons rappelé dans la section précédente, on détermine l'architecture du PMC adapté au problème par sélection de modèles. Nous procéderons de même pour les PMCF : il s'agira donc ici de choisir, par une technique de sélection de modèles, le nombre de neurones p ou plus généralement l'architecture du PMCF (cf la fonction 3.5).

3. Mise en œuvre pratique

La mise en œuvre informatique du PMC fonctionnel est délicate en raison de trois problèmes.

Tout d'abord, certains paramètres du modèle proposé sont des fonctions. Or, la manipulation informatique de fonctions est difficile et on doit se contenter de combinaisons de fonctions élémentaires ou de fonctions discrétisées.

De plus, le neurone fonctionnel s'appuie sur un calcul d'intégrales dont la réalisation informatique peut poser des problèmes.

Enfin, il est très rare de disposer de données sous une forme fonctionnelle au sens mathématique du terme (par exemple une définition en terme de fonctions connues). En général, on dispose au contraire pour chaque fonction considérée d'une discrétisation de celle-ci. Ainsi la fonction g^i est-elle connue par l'intermédiaire d'un m^i-uplets de couples entrée/sortie, les $(x^i_j, y^i_j)_{1\leq j\leq m^i}$. Pour prendre en compte le bruit d'observation, on suppose que $y^i_j = g^i(x^i_j) + \epsilon^i_j$, où ϵ^i_j désigne un bruit centré indépendant des x^i_j. On suppose que les x^i_j sont choisis de façon aléatoire selon une loi P_X, ce qui constitue une extension importante par rapport au modèle déterministe utilisé en général en ADF. Nous offrons ainsi un modèle plus réaliste pour les situations dans lesquelles la discrétisation des fonctions n'est pas la même pour chaque fonction. C'est le cas par exemple pour certaines données médicales (cf [JAM 00]). En effet, bien que les patients soient suivis de la même façon, les conditions pratiques de ce suivi font qu'ils ne peuvent pas être observés aux mêmes instants (par rapport à leur date

d'entrée dans le programme de soins). Le nombre d'observations (m^i) est donc différent pour chaque patient, de même que les instants d'observations (les x_j^i). Un autre exemple naturel est celui de séries temporelles (la quantité d'un polluant dans l'atmosphère mesurée à intervalles réguliers dans divers endroits) échantillonnées régulièrement mais provenant d'appareils de mesure fragiles. Dans ce type de situation, de nombreuses données sont manquantes, en raison de pannes sporadiques des capteurs. Bien que l'échantillonnage soit en théorie régulier, il est en pratique différent pour chaque série temporelle. On pourrait ainsi multiplier les exemples pratiques qui conduisent à envisager une discrétisation différente pour chaque fonction observée avec un choix aléatoire des points d'observation. Pour conserver une cohérence à la discrétisation, nous devons cependant imposer aux points d'observations une distribution commune, notée P_X. De plus, nous intégrons cette distribution en choisissant comme mesure $\mu = P_X$ pour les intégrales qui interviennent dans la définition du PCMF.

Nous devons donc trouver une méthode permettant d'adapter le PCMF théorique proposé dans la section précédente aux contraintes qui viennent d'être présentées. Dans la suite de cette section, nous traitons la représentation des fonctions de poids puis nous détaillons deux solutions possibles aux problèmes de la représentation des données et du calcul des intégrales. Nous nous contenterons d'étudier le calcul effectué par un neurone fonctionnel, le calcul réalisé par le PMC en découlant naturellement.

3.1. *Fonctions de poids*

Les fonctions de poids sont représentées naturellement grâce à une combinaison de fonctions simples. On peut par exemple réaliser une combinaison linéaire ce qui revient à écrire les fonctions de poids sous la forme :

$$f(x) = \sum_{k=1}^{l} w_k \phi_k(x), \tag{5}$$

où les $(\phi_k)_{1 \leq k \leq l}$ sont des fonctions faciles à calculer informatiquement. On peut par exemple choisir pour les ϕ_k une base de B-splines, d'ondelettes ou de fonctions trigonométriques. Dans tous les cas, on remplace le paramètre fonctionnel du neurone par un vecteur de paramètres numériques, les $(w_k)_{1 \leq k \leq l}$.

On peut aussi utiliser une combinaison plus complexe et surtout non linéaire par rapport à ses paramètres, comme par exemple un PMC numérique. On écrit alors les fonctions de poids sous la forme :

$$f(x) = F(w, x), \tag{6}$$

où w désigne le vecteur de paramètres du modèle non linéaire choisi.

En pratique, il est possible de comparer plusieurs solutions pour la représentation des fonctions de poids par une technique de sélection de modèles appliquée aux différents PMCFs obtenus (cf la section 3.5). On peut ainsi adapter la base fonctionnelle aux données étudiées.

3.2. *Approche directe*

L'implémentation la plus simple du PMCF théorique consiste à approcher les intégrales par des moyennes empiriques. Comme indiqué en introduction de la section 3, la fonction g^i est connue sous la forme d'une liste de m^i couples d'observations $(x_j^i, y_j^i)_{1 \leq j \leq m^i}$, les x_j^i étant distribués selon P_X. On peut alors approcher $\int f g^i dP_X$ par $\frac{1}{m^i} \sum_{j=1}^{m^i} y_j^i f(x_j^i)$ (grâce à la loi forte des grands nombres). On voit ici l'intérêt pratique d'avoir défini le neurone fonctionnel pour une mesure μ quelconque : on peut ainsi l'adapter à la discrétisation observée en posant $\mu = P_X$.

Considérons une fonction de poids sous la forme générale $F(w, x)$. Un neurone fonctionnel basé sur cette représentation peut être implémenté informatiquement en associant à la liste des m^i couples d'observations la valeur numérique suivante :

$$T\left(b + \frac{1}{m^i} \sum_{j=1}^{m^i} y_j^i F(w, x_j^i)\right)$$

Intuitivement, on a donc adapté le neurone numérique en lui permettant de traiter des listes d'observations de taille variable (car m^i dépend de i). La nature de F intervient dans l'optimisation des calculs. En effet, si F est non linéaire (par rapport à w), on ne peut pas simplifier le calcul de l'expression. Par contre, si F est linéaire en w, de la forme $F(w, x) = \sum_{k=1}^{l} w_k \phi_k(x)$, on peut calculer une fois pour toute les grandeurs suivantes (au nombre de l) :

$$\gamma_k = \frac{1}{m^i} \sum_{j=1}^{m^i} y_j^i \phi_k(x_j^i)$$

Le neurone calcule alors :

$$T\left(b + \sum_{k=1}^{l} w_k \gamma_k\right)$$

En général, l est petit devant m^i et cette technique permet donc de réduire considérablement le temps de calcul. De plus, l est une constante du modèle, alors que m^i dépend de la fonction. Avec ce mode de calcul, on peut donc se ramener à un nombre constant d'entrées pour le réseau de neurones. Il est ainsi possible de soumettre les données pré-traitées à une implémentation informatique classique des PMC.

L'intérêt principal de cette approche réside dans sa souplesse, puisqu'elle permet de représenter les fonctions de poids de façon non linéaire.

3.3. *Approche par projection*

En ADF, les fonctions observées sont en général manipulées par l'intermédiaire de représentants réguliers (cf [RAM 97]). Plus précisément, on utilise les observations d'une fonction pour déterminer une approximation de celle-ci simple à manipuler. Même s'il est possible en théorie d'utiliser une méthode d'approximation non linéaire (comme un PMC classique), cela n'a pas beaucoup d'intérêt pratique. Tout d'abord, construire une approximation non linéaire est coûteux en temps de calcul. De plus, la représentation obtenue est en général plus difficile à manipuler qu'une représentation linéaire. C'est pourquoi les méthodes classiques de l'ADF travaillent en général par projection des fonctions observées sur une base fonctionnelle tronquée, par exemple une base de B-splines, une base d'ondelettes ou encore une base de fonctions trigonométriques. La seule contrainte induite par cette technique est qu'on doit se restreindre à des fonctions dans $L^2(\mu)$.

Étant donnée g^i, on cherche donc les coefficients $\alpha_k(g^i)$ qui minimisent :

$$\left\| g^i - \sum_{k=1}^{l} \alpha_k(g^i)\psi_k \right\|^2,$$

les $(\psi_k)_{1\leq k\leq l}$ étant les fonctions de base considérées. Comme dans la section précédente, on approche ce problème en posant $\mu = P_X$ et en utilisant la loi des grands nombres pour dire que la norme proposée ci-dessus s'approche par :

$$\frac{1}{m^i}\sum_{j=1}^{m^i}\left(y_j^i - \sum_{k=1}^{l}\alpha_k(g^i)\psi_k(x_j^i)\right)^2$$

On remplace donc la liste de m^i couples d'observations $(x_j^i, y_j^i)_{1\leq j\leq m^i}$ par la fonction $\sum_{k=1}^{l}\alpha_k(g^i)\psi_k$. Cette méthode ne résout qu'en partie les problèmes d'implémentation du neurone fonctionnel. Il reste en effet à calculer l'intégrale $\int F(w,x)(\sum_{k=1}^{l}\alpha_k(g^i)\psi_k(x))dP_X(x)$. Par linéarité, il suffit de calculer les $\int F(w,x)\psi_k(x)dP_X(x)$.

L'emploi d'une technique de type Monte Carlo est envisageable pour ce calcul, puisqu'on connaît des listes de points d'observations (les x_j^i) qui sont par hypothèse distribués selon P_X. Dans cette optique, le seul intérêt de l'étape de projection est la régularisation que celle-ci peut apporter (ce qui permet de lutter contre le bruit d'observations, les ϵ_j^i). Une autre solution est de choisir pour F une fonction linéaire de la forme $F(w,x) = \sum_{s=1}^{r} w_s\phi_s(x)$. On est ainsi ramené aux calculs des $\int \psi_k\phi_s dP_X$. Ces intégrales se calculent une fois pour toute (par une méthode de Monte Carlo pour respecter la mesure d'intégration P_X). Le neurone calcule alors :

$$T\left(b + \sum_{s=1}^{r} w_s\left(\sum_{k=1}^{l}\alpha_k(g^i)\int\psi_k\phi_s dP_X\right)\right)$$

L'avantage par rapport au cas de F quelconque est double, comme dans la section précédente : certains calculs lourds sont faits au préalable à l'estimation des paramètres du réseau de neurones et la taille variable des observations disparaît car le neurone traite en fait le vecteur (de taille fixe) suivant :

$$\left(\sum_{k=1}^{l}\alpha_k(g^i)\int\psi_k\phi_1 dP_X, \ldots, \sum_{k=1}^{l}\alpha_k(g^i)\int\psi_k\phi_r dP_X\right)$$

3.4. *Estimation des paramètres*

Les deux techniques proposées, combinées à la représentation des fonctions de poids par une fonction donnée, conduisent à des versions du PMC fonctionnel qui peuvent être implémentées informatiquement et qui n'utilisent pas de paramètres fonctionnels.

Pour fixer les idées, considérons l'approche directe appliquée au cas d'une représentation des fonctions de poids par l'intermédiaire d'une unique fonction F. On suppose connues n fonctions, les $(g^i)_{1\leq i\leq n}$. La fonction g^i est décrite par la liste de couples $(x^i_j, y^i_j)_{1\leq j\leq m^i}$. On souhaite appliquer le PMC fonctionnel simple décrit par l'équation 3 à ces données. Avec l'implémentation choisie, l'image par le PMC fonctionnel de la fonction i est :

$$H(\theta, g^i) = \sum_{k=1}^{p} a_k T\left(b_k + \frac{1}{m^i}\sum_{j=1}^{m^i} y^i_j F(w_k, x^i_j)\right) + c \tag{7}$$

Dans cette équation, θ regroupe l'ensemble des paramètres du PMC fonctionnel sous forme d'un unique vecteur.

Dans un problème de régression, chaque observation g^i est associée à une grandeur à prédire, notée ici s^i. L'estimation des paramètres du PMC consiste à trouver une valeur de θ (i.e des valeurs des w_k, des a_k, des b_k et de c) telle que $H(\theta, g^i) \simeq s^i$ pour tout i. Plus précisément, pour une architecture fixée pour le PMCF, le modèle de régression retenu est le suivant :

$$s^i = H(\theta, g^i) + \epsilon^i, \tag{8}$$

où ϵ^i désigne un bruit centré. On obtient ainsi une version exploitable numériquement du modèle de régression sur variable fonctionnelle de l'équation 4. Pour estimer les paramètres de ce modèle, il suffit comme pour un PMC classique, ou plus généralement un modèle paramétrique, de choisir un critère d'erreur d et de chercher les paramètres qui minimisent :

$$\mathcal{E}(\theta) = \sum_{i=1}^{n} d(H(\theta, g^i), s^i) \tag{9}$$

En pratique, on adapte d au problème, en considérant l'erreur quadratique dans le cas de la régression et l'entropie croisée pour la discrimination. Les paramètres optimaux sont obtenus grâce à un algorithme classique d'optimisation (par exemple un quasi-Newton, cf [PRE 92, CIA 00]). Plus généralement, pour une architecture différente de PMCF ou pour l'approche par projection, la forme du modèle de régression ne change pas : seule la fonction H est modifiée pour intégrer le mode de calcul et l'architecture retenus.

3.5. *Sélection de modèles*

La définition du PMCF conduit à une classe de modèles dans laquelle on doit choisir celui qui est le plus adapté aux données étudiées. Comme nous allons le voir dans la section 3.7, nous pouvons nous limiter en pratique à un PCMF comportant deux couches. Le nombre d'entrées de la première couche et le nombre de neurones de la deuxième couche sont alors imposés par le problème étudié, comme dans le cas classique. Déterminer l'architecture du PMCF revient donc à choisir le nombre de neurones de la première, p dans l'équation 7. Nous pouvons aussi choisir le mode de représentation des fonctions de poids, par exemple le nombre de fonctions de base dans le cas d'une représentation linéaire, ainsi que le mode d'implémentation choisi (approche directe ou par projection). En pratique, nous devons donc comparer la qualité de plusieurs modèles paramétriques de régression correspondant aux divers choix possibles pour les éléments étudiés.

La comparaison de modèles n'a rien de spécifique aux réseaux de neurones ou aux données fonctionnelles, et on peut donc utiliser des techniques classiques. Nous avons retenu une méthode de validation croisée pour estimer les performances d'un modèle. Plus précisément, nous supposons que les données à modéliser sont toujours réparties en deux ensembles : les données d'apprentissage et celles de test. Les données de test sont exclusivement utilisées pour estimer les performances du modèle finalement retenu et n'interviennent à aucun autre moment. Les données d'apprentissage sont découpées aléatoirement en k blocs de tailles similaires. Pour chaque bloc, on estime les paramètres du modèle sur les $k-1$ autres blocs comme indiqué dans la section 3.4, puis on calcule les performances du modèle sur le bloc étudié. La qualité globale du modèle est obtenue en combinant les performances obtenues sur chaque bloc.

A chaque choix architectural pour le PMCF est ainsi associé un indice de performance. On compare les différentes possibilités grâce à cet indice, ce qui permet de choisir un modèle particulier. Une fois cette procédure achevée, les paramètres du modèle retenu sont estimés sur l'ensemble des données d'apprentissage, puis le modèle est évalué sur les données de test.

Cette procédure (extrêmement classique) permet ainsi de choisir automatiquement un modèle paramétrique adapté aux données, parmi l'ensemble des modèles de type PMCF applicables au

problème étudié. Bien entendu, il est impossible en pratique d'explorer exhaustivement cette classe infinie et il faut donc choisir des plages de variation pour les caractéristiques des modèles retenus. Par exemple, on limite le nombre de neurones de la première couche à une valeur maximale déterminée de façon heuristique. De la même façon, on fixe en général une classe de bases pour la représentation de fonction (par exemple des B-splines de degré 3) ainsi qu'une plage pour le nombre de fonctions dans chaque base étudiée.

Notons pour conclure cette section que d'autres méthodes sont envisageables pour la sélection de modèles, en particulier le *bootstrap*.

3.6. *Mise en œuvre informatique*

Comme nous l'avons rappelé dans la section 3.2, un des avantages des PMC classiques est leur parcimonie. En pratique en effet, un modèle quelconque (paramétrique ou non) doit être implémenté sur un ordinateur, ce qui impose des contraintes en terme de puissance de calcul et de mémoire disponible, surtout si l'ordinateur est simple (par exemple dans un appareil photo, dans un assistant personnel ou dans un téléphone portable). L'estimation des paramètres du modèle ou la construction de ce dernier peut être réalisée une fois pour toute sur un ordinateur puissant dédié à ce genre de modélisation, mais la mise en œuvre finale se fera sur le terminal adapté à l'application.

L'avantage des PMC dans cette situation est qu'ils permettent la construction de modèles très peu coûteux en temps de calcul et en occupation mémoire. En effet, après la phase de sélection de modèles (très coûteuse en temps de calcul en général), le PMC retenu utilise très peu de paramètres et résume de cet fait de façon très concise les données étudiées. Au contraire, les modèles non paramétriques à noyaux sont très peu concis. Ils nécessitent en effet le stockage de l'intégralité des données d'origine (éventuellement sous une forme simplifiée). En contrepartie, la phase de sélection de modèles est réduite à sa plus simple expression (en général, on doit se contenter de régler le paramètre de largeur du noyau) et la construction du modèle est donc très rapide.

3.7. *Propriétés théoriques*

Nous résumons ici des propriétés théoriques énoncées de façon plus formelle dans [ROS 05, CON 02]. Nous avons en fait étendu au cas du PMCF deux propriétés fondamentales des PMC (cf [LES 93], [HOR 93], [STI 99] et [WHI 89] pour le cas des PMC classiques) :

1. nous avons montré que le PMCF est un approximateur universel, c'est-à-dire qu'on peut approcher arbitrairement bien une fonction continue d'un compact d'un espace fonctionnel $L^p(\mu)$ dans $\mathbb{R}$ avec un PCMF à deux couches (nous étendons les résultats de [STI 99]) ;

2. nous avons montré que l'estimation des paramètres d'un PMCF d'architecture fixée à partir d'un échantillon de fonctions discrétisées est consistante au sens où les valeurs des paramètres obtenues de cette manière convergent presque sûrement vers les paramètres optimaux théoriques.

4. Exemples d'applications

4.1. *Protocole expérimental*

Nous présentons dans cette section les résultats de l'application de nos deux méthodes neuronales (l'approche directe et l'approche par projection) à des données simulées (les vagues de Breiman) et à des données réelles (un problème de spectrométrie). Notre but est avant tout de comparer nos méthodes aux méthodes classiques et fonctionnelles étudiées dans [FER 03a]. Les auteurs mettent en œuvre dans cet article une méthode originale qui généralise le principe des estimateurs à noyaux au cas fonctionnel (cf aussi [FER 02b] et [FER 02a]). Ils comparent cette méthode à de nombreuses techniques adaptées aux données fonctionnelles comme l'analyse discriminante pénalisée (*Penalized Discriminant Analysis*, cf [HAS 95]) et la régression PLS multiple [MAR 89], ainsi qu'à des techniques plus classiques comme les arbres de régression [BRE 84] et l'analyse discriminante "souple" proposée dans [HAS 94].

Nous avons mis en œuvre sur ces données trois modèles neuronaux : un PMC classique appliqué de façon naïve aux données fonctionnelles considérées comme des observations vectorielles (ce qui

est possible car les fonctions étudiées ici sont discrétisées de façon régulière), un PMC fonctionnel utilisant l'approche directe et un PMC fonctionnel utilisant l'approche par projection.

Comme les données sur lesquelles nous travaillons sont des fonctions de $\mathbb{R}$ dans $\mathbb{R}$, il n'est pas utile (et même néfaste) d'utiliser une représentation non linéaire des fonctions de poids et des observations. De ce fait, nous avons opté pour une représentation par une base de B-splines. Le choix du PMC fonctionnel demande alors le réglage de deux paramètres architecturaux : le nombre de neurones dans la première couche (en accord avec le résultat d'approximation universelle, nous nous sommes limités à des PMC à deux couches) et le nombre de noeuds dans la base de B-splines (que celle-ci soit utilisée pour une représentation des observations par projection ou pour la représentation des fonctions de poids dans l'approche directe).

L'estimation des modèles neuronaux est assez délicate car ils sont sensibles au phénomène de sur-apprentissage : le réseau de neurones apprend le bruit et les performances sur l'ensemble de test sont bien moins bonnes que celles obtenues sur l'ensemble d'apprentissage. Pour lutter contre ce phénomène, nous avons utilisé une technique classique de pénalisation, le *weight decay* [BIS 95] qui est une reformulation neuronale des techniques liées à la régression *ridge* [HOE 70b, HOE 70a, CAZ 75]. Il s'agit en effet de pénaliser les grandes valeurs des poids du PMC en ajoutant à l'erreur définie par l'équation 9 le terme $\delta\|\theta\|^2$. La méthode nécessite donc le réglage d'un paramètre, le taux de régularisation δ. Notons que cette technique donne en général de meilleurs résultats que celle de l'arrêt prématuré de l'apprentissage (*early stopping*) utilisée par exemple dans [ZHU 97]. Nous aurions pu choisir un terme de pénalisation plus évolué, comme par exemple la dérivée seconde de la fonction calculée par le PMC (cf [BIS 93]), mais les résultats obtenus montrent que la solution simple retenue donne de très bons résultats sans demander un temps de calcul important et une mise en œuvre complexe.

Comme indiqué dans la section 3.5, les choix architecturaux (nombre de neurones dans la première couche et nombre noeuds dans la base de B-spline), ainsi que le taux de régularisation sont déterminés par validation croisée. Pour limiter les temps de calcul, nous avons choisi de d'utiliser un découpage en cinq blocs de l'échantillon d'apprentissage. Ce choix a été seulement guidé par des considérations

informatiques. Des expériences réduites (portant sur un seul découpage des données en apprentissage et test) ont confirmé que le nombre de blocs (pour des valeurs de l'ordre de 5) n'avait pas d'effet notable sur le choix de l'architecture.

Notons pour finir que les résultats indiqués dans cette section correspondent toujours aux performances sur un ensemble de test totalement distinct de l'échantillon d'apprentissage.

4.2. *Données simulées : les vagues de Breiman*

Comme dans [FER 03a], nous avons réalisé une première expérience de mise en œuvre de nos méthodes sur des données simulées, les vagues introduites par Breiman dans [BRE 84]. Il s'agit d'un problème à trois classes. Chaque classe de fonctions est engendrée par combinaison linéaire à partir de trois vagues maîtresses, des fonctions continues définies sur l'intervalle $[1, 21]$ par les équations suivantes :

$$h_1(t) = \max(6 - |t - 11|, 0) \tag{10}$$

$$h_2(t) = h_1(t - 4) \tag{11}$$

$$h_3(t) = h_1(t + 4) \tag{12}$$

Les fonctions à classer sont définies de la façon suivante :

$$x(t) = uh_1(t) + (1 - u)h_2(t) \text{ pour la classe 1,} \tag{13}$$

$$x(t) = uh_1(t) + (1 - u)h_3(t) \text{ pour la classe 2,} \tag{14}$$

$$x(t) = uh_2(t) + (1 - u)h_3(t) \text{ pour la classe 3,} \tag{15}$$

où u est un réel de $]0, 1[$. Dans [BRE 84], chaque fonction est discrétisée en un vecteur de $\mathbb{R}^{21}$ par une discrétisation uniforme sur $[1, 21]$. Pour renforcer l'aspect fonctionnel de ces données, nous travaillons comme [FER 03a] en les discrétisant en 101 points régulièrement répartis dans l'intervalle $[1, 21]$.

Nous engendrons les données exactement comme dans [FER 03a]. L'ensemble d'apprentissage est constitué de 150 fonctions par classe. Le paramètre u est choisi aléatoirement uniformément dans

$]0,1[$ (indépendamment pour chaque fonction) et un bruit gaussien (centré et de variance unité) est ajouté à chaque observation. La figure 2 représente 20 vagues pour chaque classe. Les données de test sont engendrées de la même façon, mais avec 250 fonctions dans chaque classe.

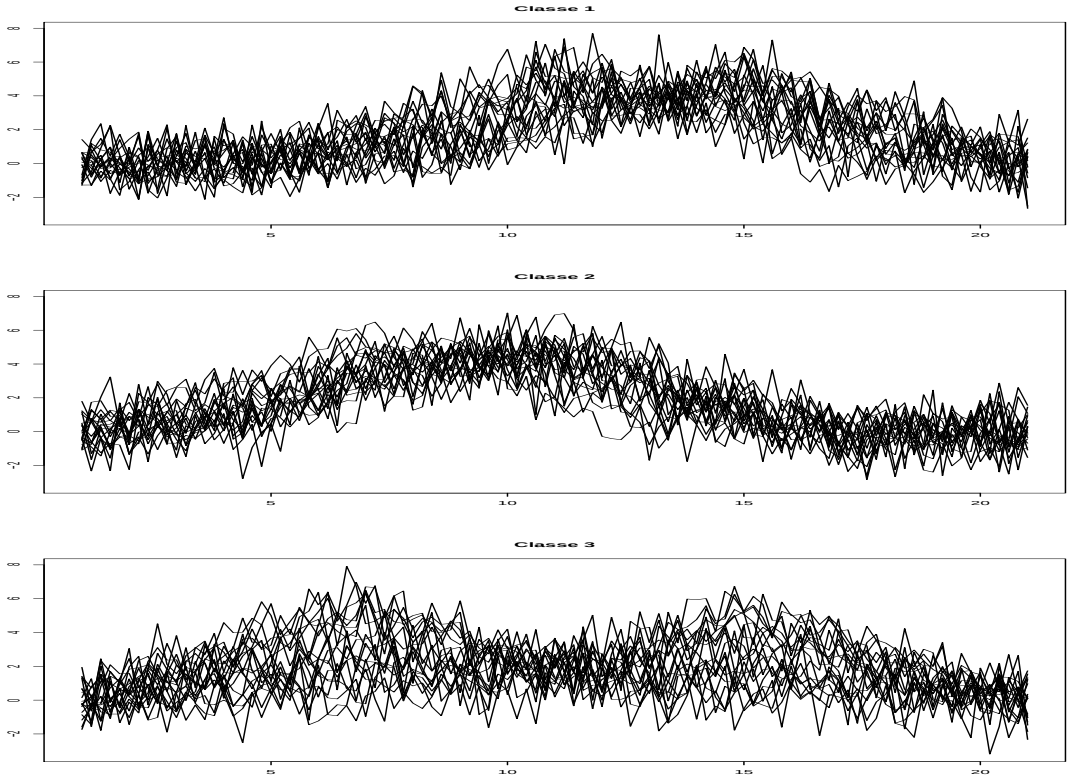

FIG. 2 – 20 vagues pour chaque classe

La table 1 donne les résultats obtenus par les trois approches neuronales. Le taux d'erreur de classement correspond à une moyenne sur 50 échantillons de test, alors que l'écart type donne une idée de la dispersion de ce taux d'erreur (cette présentation des résultats permet une comparaison directe avec ceux de [FER 03a]). Chaque échantillon de test est associé à un échantillon d'apprentissage.

Les résultats obtenus sont très satisfaisants. Tout d'abord, les approches fonctionnelles donnent de meilleurs résultats que l'approche classique. Les différences de performance entre les méthodes sont en réalité plus importantes que le seul taux d'erreur moyen ne l'indique. On peut en effet comparer les performances échantillon par échantillon : on constate alors que les approches fonctionnelles sont

Méthode	Taux d'erreur moyen	écart type
Classique	0.098	0.013
Directe	0.065	0.0096
Projection	0.072	0.011

TAB. 1 – Taux d'erreur sur l'échantillon de test pour les vagues de Breiman

toujours meilleures que l'approche classique. De plus, l'approche directe donne un meilleur résultat que l'approche par projection dans 38 tirages sur 50.

De plus, les méthodes fonctionnelles conduisent à une représentation parcimonieuse des données. En effet, l'approche directe utilise en moyenne un modèle à 44 paramètres, contre 406 paramètres pour l'approche classique (et 36 seulement pour l'approche par projection). Les architectures retenues sont différentes pour chaque tirage. Cependant, dans plus de la moitié des tirages, les trois approches sélectionnent un modèle à 3 neurones dans la première couche (il y a toujours un unique neurone dans la deuxième couche). La parcimonie des approches fonctionnelles vient en fait de la représentation par B-splines. En effet, le modèle sélectionné comporte en général peu de fonctions de base (moins de 10), ce qui réduit d'autant le nombre d'entrées du PMCF et donc le nombre de paramètres.

Enfin, les PMCF se comportent mieux que toutes les méthodes explorées dans [FER 03a]. La meilleure méthode étudiée dans [FER 03a] atteint en effet un taux d'erreur moyen de 0.072 (avec un écart type de 0.012). La différence de performance n'est pas significative, mais elle montre que les PMCF sont une technique compétitive pour le traitement de données fonctionnelles, comme l'application en spectrométrie le confirme. De plus, la meilleure méthode proposée dans [FER 03a] demande un stockage important. En effet, il s'agit d'une méthode à noyaux, ce qui implique un stockage de toutes les données d'apprentissage pour pouvoir classer une nouvelle courbe. Les fonctions sont comparées dans [FER 03a] par l'intermédiaire de leur projection sur un espace vectoriel dont la base est construite par régression PLS. Le nombre optimal de fonctions de base est déterminé par validation croisée et vaut ici 3. Chaque fonction de l'ensemble d'apprentissage peut donc être réduite à 3 coordonnées, soit au total $450 \times 3 = 1350$ nombres réels. De plus, pour classer une nouvelle fonc-

tion, il faut calculer ses coordonnées sur la base choisie, ce qui impose le stockage des coordonnées des fonctions qui la forment dans la base d'origine. Nous avons donc 101 coordonnées pour chacune des 3 fonctions de base, soit un total de 1653 nombres réels. La méthode de Ferraty et Vieu réduit considérablement le volume des données d'apprentissage ($450 \times 101 = 45450$ nombres réels), ce qui est remarquable pour une méthode non paramétrique, mais reste beaucoup moins parcimonieuse que les approches neuronales. En pratique, le volume de stockage très important peut poser des problèmes si on envisage l'exploitation de la méthode sur un composant de puissance limitée.

En conclusion, l'expérience réalisée sur les vagues de Breiman donne des résultats très satisfaisants. Les performances obtenues sont très bonnes et comparables à celles produites par les meilleures méthodes étudiées dans [FER 03a]. De plus, l'approche neuronale fonctionnelle se comporte bien mieux qu'un PMC classique, à la fois en terme de performances mais aussi en terme de parcimonie du modèle obtenu. Cette caractéristique est particulièrement intéressante car elle limite les ressources nécessaires à une implémentation du modèle après apprentissage, en particulier par rapport à une approche non paramétrique. De plus, la construction du PMCF optimal demande un temps de calcul similaire à celle du PMC classique : le gain en performances et en parcimonie n'entraîne donc pas de pénalités.

4.3. *Application en spectrométrie*

4.3.1. *Données brutes*

Nous travaillons sur des données spectrométriques utilisées pour classer des échantillons selon une propriété physico-chimique qui n'est pas accessible directement (et demande donc une analyse spécifique). Les données utilisées ont été obtenues grâce à un *Tecator Infratec Food and Feed Analyzer*[1] travaillant dans le proche infra-rouge (longueurs d'onde comprises entre 850 et 1050 nanomètres). La mesure est effectuée par transmission à travers un échantillon de viande finement hachée qui est ensuite analysé par un procédé chimique pour déterminer son taux de graisse. Les spectres obtenus correspondent à l'absorbance ($-\log_{10}$ de la transmitance mesurée par l'appareil) pour 100 longueurs

1. Les données sont disponibles à l'URL `http://lib.stat.cmu.edu/datasets/tecator`

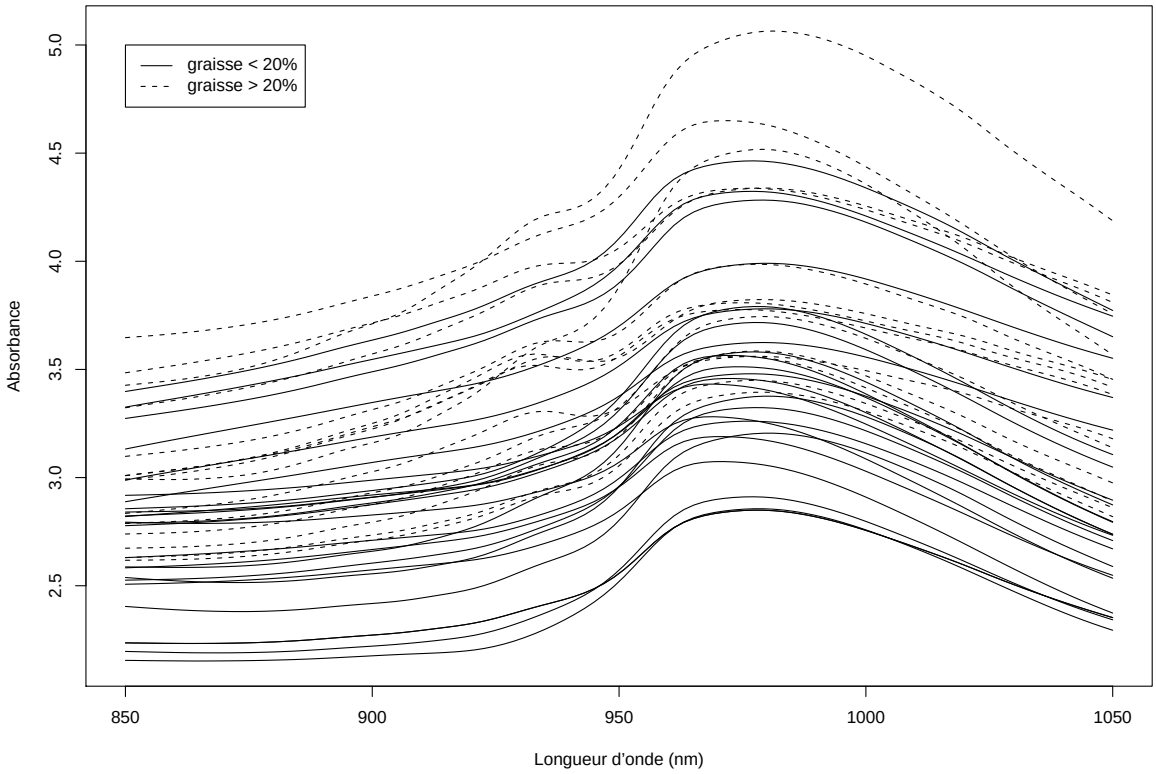

FIG. 3 – 40 spectres

d'onde régulièrement réparties entre 850 et 1050 nm. Les échantillons de viande sont répartis en deux classes, selon qu'ils contiennent plus ou moins de 20 % de graisse (on a 77 spectres correspondant à plus de 20 % de graisse et 138 avec moins de 20 % de graisse). Le problème est alors de discriminer les spectres afin d'éviter l'analyse chimique, coûteuse et longue. La figure 3 représente 40 spectres choisis aléatoirement dans les 215 échantillons.

Pour nous confronter aux résultats de [FER 03a], nous avons repris le protocole expérimental proposé par les auteurs. Le jeu de données d'origine est en effet séparé aléatoirement en un ensemble d'apprentissage dont la taille a été fixée à 160, utilisé pour estimer le modèle et un ensemble de test de taille 55, utilisé pour estimer les performances du modèle obtenu. Nous répétons cette séparation aléatoire 50 fois et comparons ensuite les méthodes en fonction de la médiane du taux de mauvais classement. Notons que nous n'avons pas cherché à respecter les proportions des deux classes dans chacun des deux échantillons d'apprentissage et de test.

Méthode	Premier quartile	Médiane	Moyenne	Troisième quartile	Maximum
Classique	0	0.018	0.019	0.036	0.054
Directe	0.018	0.036	0.028	0.036	0.091
Projection	0	0.018	0.018	0.036	0.054

TAB. 2 – Taux d'erreur sur l'échantillon de test pour les données spectrométriques

La table 2 résume les taux de mauvais classement obtenus par les trois techniques neuronales étudiées. On constate que les taux d'erreur obtenus sont très bons puisque dans plus de la moitié des tirages (répartitions aléatoires entre l'ensemble d'apprentissage et l'ensemble de test), la méthode fonctionnelle par projection donne moins de 2 % d'erreur de classement. Cependant, cette méthode n'est que marginalement meilleure que l'approche neuronale classique (en fait, une comparaison tirage par tirage indique que l'approche par projection donne de meilleures performances sur 30 tirages parmi 50 et des performances égales sur 13 tirages). De plus, l'approche directe donne de moins bons résultats que les deux autres méthodes. En fait, le véritable intérêt des approches fonctionnelles réside ici dans leur parcimonie car la méthode par projection utilise en moyenne 69 paramètres numériques contre 288 pour l'approche classique, ce qui limite les besoins en stockage et les temps de calcul en phase d'exploitation. Comme dans les expériences sur les vague de Breiman (section 4.2), les architectures optimales dépendent du tirage. On constate cependant que dans plus de la moitié des tirages, le modèle optimal utilise 2 ou 3 neurones dans la première couche (pour les trois approches). Comme pour les vagues de Breiman, c'est le recours à une représentation par B-splines qui réduit le nombre d'entrées des PMCF et les rend parcimonieux. Les spectres sont en effet efficacement représentés par 15 à 20 fonctions de base, alors que le PMC classique utilise 100 entrées.

La comparaison avec les résultats rapportés dans [FER 03a] montre que les meilleures approches neuronales sont très performantes puisque la meilleure méthode étudiée dans [FER 03a] obtient un taux d'erreur médian d'environ 2.2 %. Comme pour les vagues de Breiman, la meilleure méthode compare les fonctions par l'intermédiaire d'une projection sur un sous-espace fonctionnel construit en utilisant une régression PLS multivariée. D'après [FER 03a], une approche non paramétrique

fonctionnelle pure, c'est-à-dire comparant directement les fonctions sans projection préalable, donne de mauvais résultats (de l'ordre de 20 % d'erreur de classement en moyenne), vraisemblablement pour des raisons de décalages entre les spectres, comme nous le verrons dans la section suivante.

Notons de plus que la meilleure méthode de [FER 03a] nécessite le stockage de 1300 nombres réels. Comme nous l'avons déjà indiqué dans la section précédente, les méthodes à noyaux imposent en effet le stockage de l'intégralité des données d'apprentissage. Comme pour les vagues de Breiman, la meilleure méthode non paramétrique utilise une projection sur une base déterminée par régression PLS. Cette base comporte ici 5 fonctions et les données d'apprentissage peuvent donc être stockées sous forme de $160 \times 5 = 800$ nombres réels. On doit aussi conserver les coordonnées des 5 fonctions de base dans la base d'origine, soit 500 autres nombres réels. Comme pour les vagues de Breiman, la méthode de Ferraty et Vieu réduit considérablement le volume des données d'apprentissage ($160 \times 100 = 16000$ nombres réels), mais reste beaucoup moins parcimonieuse que les approches neuronales.

4.3.2. *Pré-traitements fonctionnels*

Cette première expérience peut paraître décevante car même si elle montre l'efficacité des techniques neuronales, elle ne donne pas un clair avantage à l'approche fonctionnelle, excepté pour la parcimonie de la représentation. Cependant, l'approche fonctionnelle ne se limite pas à une technique de mise en œuvre des réseaux de neurones. En effet, elle permet de prendre en compte la nature des observations et de leurs appliquer des pré-traitements fonctionnels. Ferraty et Vieu remarquent en effet dans [FER 02b] que les spectres d'absorbance comportent souvent un effet moyen qui n'apporte aucune information. Plus précisément, la valeur moyenne de l'absorbance n'est pas un bon prédicteur du taux de graisse de l'échantillon de viande. En fait, deux spectres peuvent être très similaires en terme de forme et correspondre à des taux de graisse proches, tout en ayant des valeurs moyennes d'absorbance très différentes. La figure 4 illustre le problème avec quatre spectres correspondant à des échantillons dont la teneur de graisse est de 6.4 %.

Pour remédier à ce problème, Ferraty et Vieu proposent dans [FER 02b] et [FER 03a] de traiter les dérivées des spectres (ce qui demande une approche fonctionnelle), et montrent que la dérivée seconde est la plus informative. Plus précisément, une méthode de validation croisée utilisée dans

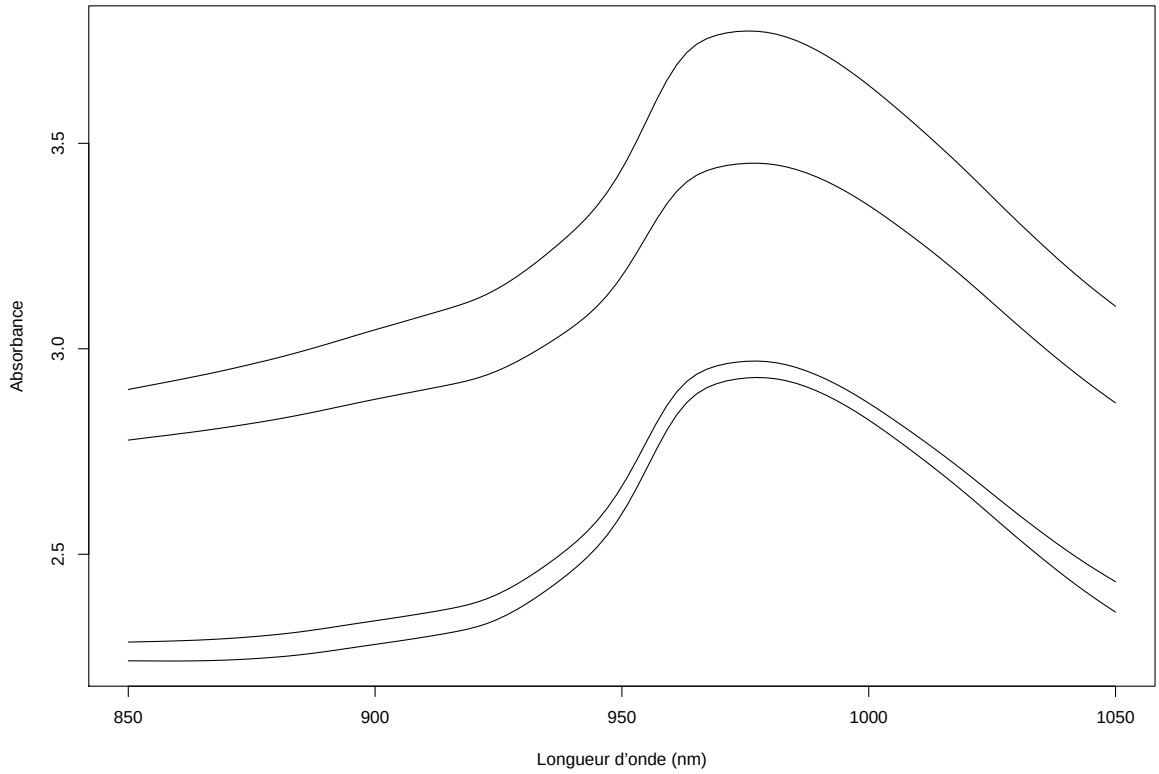

FIG. 4 – 4 spectres avec un même taux de graisse

[FER 03a] sélectionne automatiquement la dérivée seconde comme meilleure semi-norme fonctionnelle parmi les semi-normes considérées.

Comme dans [FER 03a], nous avons estimé les dérivées premières et secondes des spectres en approchant chaque spectre sur une base de B-splines (20 noeuds), en calculant les dérivées formelles des fonctions ainsi obtenues et en ré-échantillonnant les dérivées afin d'obtenir 100 observations par fonction. La figure 5 représente les dérivées secondes des 40 spectres de la figure 3. La comparaison entre les deux figures ne donne pas au premier abord un avantage particulier aux dérivées secondes. Cependant, la figure 6 montre que l'effet moyen observé sur la figure 4 disparaît : des spectres de même forme ont des dérivées secondes très proches.

Notons qu'il n'est pas possible de simplement centrer chaque courbe en soustrayant la moyenne car celle-ci apporte tout de même une information sur le taux de graisse. On constate par exemple que parmi les 138 spectres correspondant à un taux de graisse inférieur à 20 %, 75 possèdent une

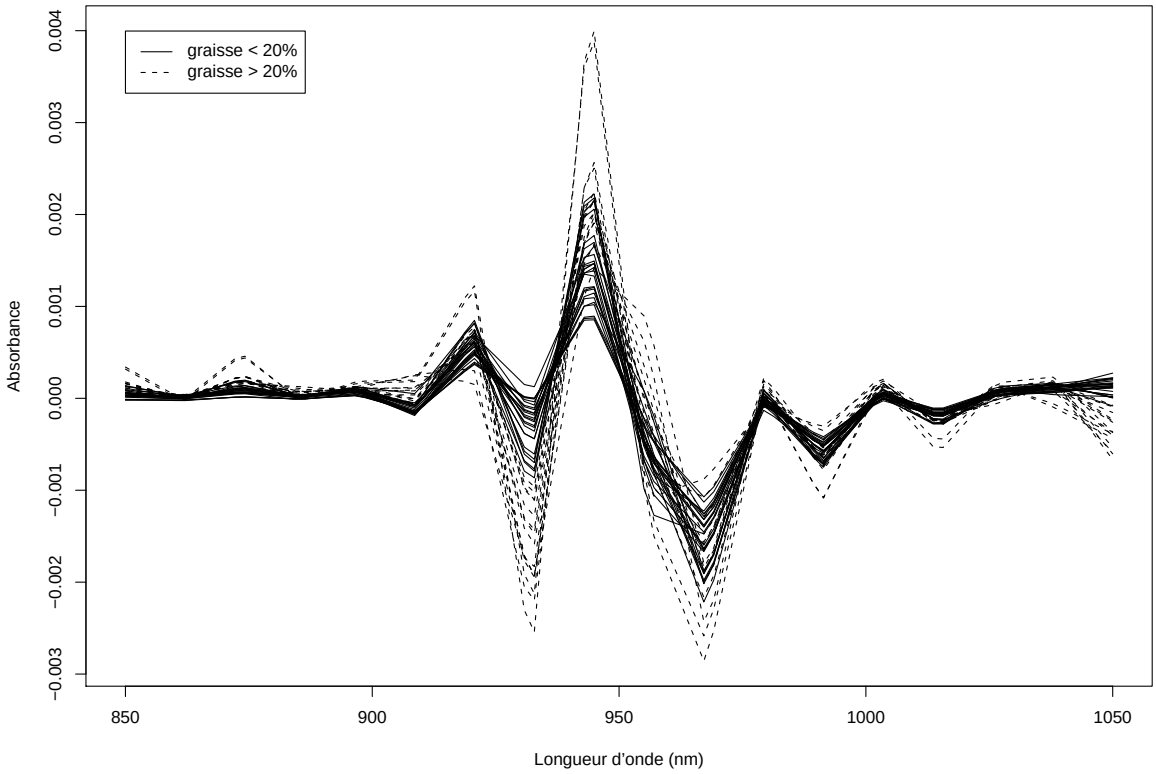

FIG. 5 – Dérivées secondes de 40 spectres

valeur moyenne d'absorbance inférieure à 3, alors que seulement 11 spectres sur les 77 correspondant à un taux de graisse supérieur à 20 % sont dans ce cas.

Méthode	Premier quartile	Médiane	Moyenne	Troisième quartile	Maximum
Classique	0	0.009	0.015	0.018	0.073
Directe	0	0	0.009	0.018	0.036
Projection	0	0.018	0.015	0.018	0.054

TAB. 3 – Taux d'erreur sur l'échantillon de test pour l'ordre de dérivation optimal des données spectrométriques

Après la dérivation, nous avons traité les fonctions obtenues exactement comme les spectres d'origine, en sélectionnant automatiquement l'ordre de dérivation. Plus précisément, nous avons inclus dans la procédure de sélection de modèles (qui détermine l'architecture du PMCF et le taux de régularisation) le choix du pré-traitement fonctionnel entre trois alternatives : données brutes,

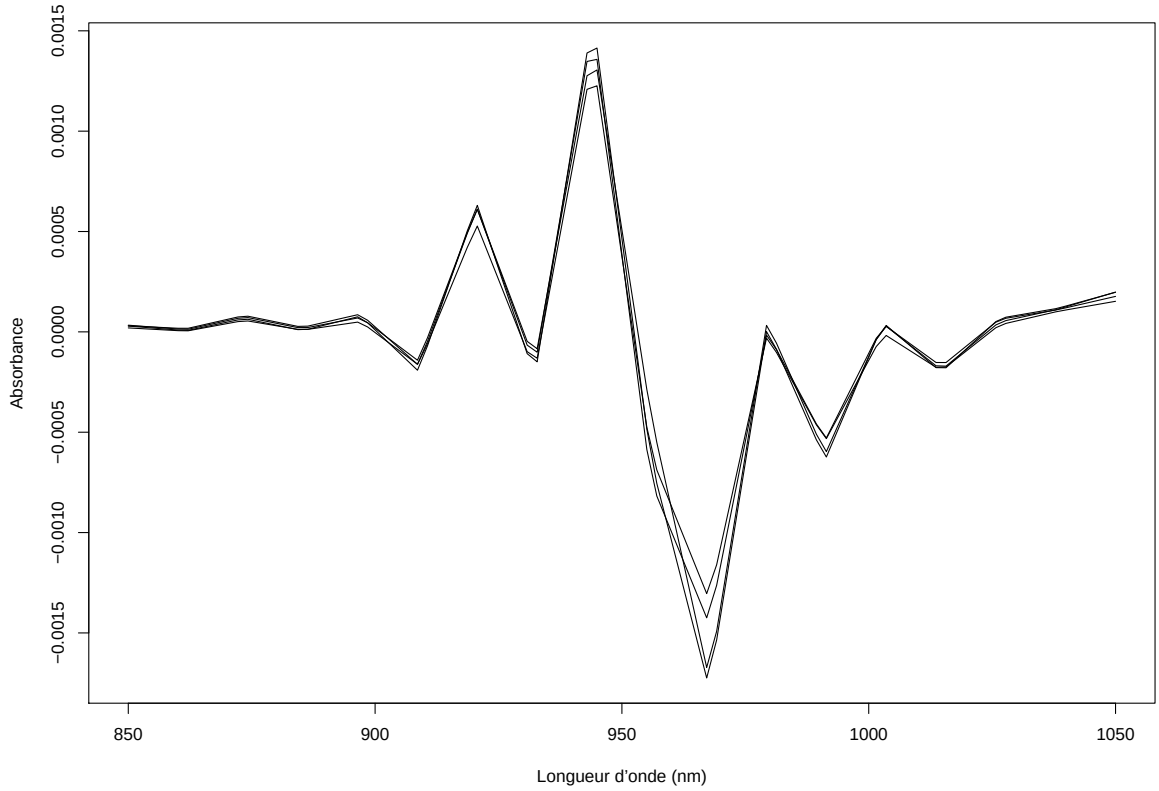

FIG. 6 – Dérivées secondes de 4 spectres avec un même taux de graisse

dérivées premières et dérivées secondes. Cette méthode transpose au cadre neuronal la procédure de sélection automatique de semi-norme utilisée dans [FER 03a].

La table 3 résume les taux de mauvais classement obtenu par les trois techniques neuronales étudiées. On constate que les performances sont améliorées, en particulier pour la méthode directe qui ne commet aucune erreur de classement pour plus de la moitié des tirages. Par contre, la méthode par projection donne à première vue des résultats décevants car elle semble moins bonne que l'approche classique. Cependant, si on compare directement les performances tirage par tirage, on constate que la méthode classique fait jeu égal avec la projection dans 25 tirages sur 50, est meilleure dans 11 tirages et moins bonne dans 14. La médiane plus faible est due au fait que l'approche classique se comporte de façon plus extrême que les méthodes fonctionnelles : elle obtient une reconnaissance parfaite dans 25 tirages (l'approche par projection se contente de 21 tirages

parfaits contre 30 pour l'approche directe), ce qui est très honorable, mais fait aussi plus de "grosses erreurs", avec un maximum à 7.27%.

Méthode	Données brutes	Dérivées premières	Dérivées secondes
Classique	5	40	5
Directe	0	6	44
Projection	5	34	11

TAB. 4 – Choix des pré-traitements optimaux

Notons d'autre part que contrairement à l'approche non paramétrique de [FER 03a], les méthodes neuronales ne sélectionnent pas systématiquement la dérivée d'ordre 2 comme étant la plus informative, comme l'indique la table 4. Celle-ci précise le nombre de tirages pour lesquels chacun des pré-traitements a été considéré comme optimal par la validation croisée.

L'approche non paramétrique proposée dans [FER 03a] donne un taux médian d'erreur de classement légèrement inférieur à 2.2 % ce qui la classe en tête des méthodes étudiées dans [FER 03a], mais derrière les approches fonctionnelles neuronales. Celles-ci restent de plus très parcimonieuses (81 paramètres numériques en moyenne pour l'approche directe, 74 pour l'approche par projection et 356 pour l'approche naïve), alors que l'utilisation d'une semi-norme basée sur la dérivée d'ordre 2 est assez pénalisante en termes de stockage pour l'approche à noyaux : contrairement au cas de la semi-norme basée sur la régression PLS, on n'observe ici aucune simplification et le stockage de l'intégralité des données est nécessaire, soit un total de 16 000 nombres réels. Comme dans les expériences précédentes, c'est le passage par une représentation par B-splines qui joue un rôle crucial dans la parcimonie des PMCF, puisqu'elle permet de passer de 100 entrées à moins de 20. Les PMC classiques ou fonctionnels optimaux comportent toujours assez peu de neurones (2 ou 3) dans la première couche (contre 1 dans la seconde couche, ce qui est imposé par la structure du problème).

Notons pour finir que l'approche classique doit être considérée en toute rigueur comme une approche mixte. En effet, seule une prise en compte du caractère fonctionnel des observations permet de tester l'effet d'une dérivation des spectres (ou d'ailleurs de toute autre transformation fonctionnelle).

Même si le traitement des données transformées est complètement classique, la transformation est elle fonctionnelle.

4.3.3. *Commentaires*

On peut tirer de ces expériences plusieurs enseignements. Tout d'abord, les approches fonctionnelles neuronales donnent des résultats très satisfaisants, au moins aussi bons qu'un modèle neuronal classique et meilleurs que ceux des méthodes concurrentes explorées dans [FER 03a]. Bien entendu, on ne peut conclure à la supériorité générale des approches neuronales sur les autres en se basant sur une unique application, d'autant plus que les différences de performances sont minimes. Les expériences montrent simplement que les PMC sont compétitifs et méritent de ce fait d'être étudiés au même titre que les autres modèles pour données fonctionnelles.

D'autre part, les modèles fonctionnels neuronaux sont parcimonieux et construisent avec peu de ressources un bon résumé des données. On pourra donc envisager leur application à des problèmes contenant de très nombreuses observations : le modèle non paramétrique fonctionnel de [FER 03a] nécessite en effet le stockage de l'intégralité des données d'apprentissage, éventuellement sous une forme simplifiée, même si celles-ci sont très redondantes, ce qui pose des problèmes d'occupation mémoire et de temps de calcul en phase d'exploitation. On retrouve dans le cadre fonctionnel les différences classiques entre les modèles neuronaux et les modèles à noyaux : les PMC proposent une représentation concise des données d'apprentissage, au prix d'une phase d'estimation très coûteuse en temps de calcul. Au contraire, les modèles à noyaux ont une phase de réglage très peu coûteuse (il s'agit en général de choisir la largeur des noyaux), au prix d'une représentation volumineuse.

Enfin, l'approche fonctionnelle permet d'appliquer des pré-traitements aux données qui tiennent compte de la nature de celles-ci ce qui améliore considérablement les performances sur les exemples étudiés.

5. Conclusion

Nous avons proposé dans cet article un modèle neuronal original adapté aux traitements des données fonctionnelles, en particulier la discrimination de telles données et plus généralement la régression sur variables explicatives fonctionnelles. Le perceptron multi-couches (PMC) fonctionnel partage avec la version numérique qu'il généralise des propriétés théoriques fondamentales, l'approximation universelle et la consistance des techniques classiques d'estimation.

Nous avons décrit deux techniques pour la mise en œuvre informatique du PMC fonctionnel, puis nous les avons appliquées à deux problèmes de discrimination : un problème artificiel (les vagues de Breiman) et un problème réel de spectrométrie. Les résultats expérimentaux montrent que les PMC fonctionnels sont des outils très satisfaisants. Les taux d'erreurs obtenus sont plus faibles que ceux atteints par d'autres méthodes (fonctionnelles ou classiques) sur les données étudiées : même si cela ne permet pas de conclure dans le cas général, les expériences menées montrent que les PMCF ne souffrent pas de défaut rédhibitoire. De plus, l'approche fonctionnelle apporte une valeur ajoutée par rapport à un simple PMC numérique, tant en terme de taux d'erreur qu'en terme d'interprétation des données et donc de pré-traitements envisageables. Enfin, la propriété de parcimonie démontrée formellement dans le cas des PMC numériques semble se retrouver expérimentalement dans le cas fonctionnel puisque les excellents taux de bon classement relevés sont obtenus par des modèles utilisant un nombre très réduit de paramètres numériques comparativement au volume des données, ce qui induit des besoins en mémoire assez réduits par rapport aux méthodes concurrentes, neuronales ou non.

Remerciements

Nous remercions les membres du groupe "STAPH[2]" (Statistique fonctionnelle) du LSP de Toulouse pour d'intéressantes discussions qui ont contribué à améliorer ce travail. Au sein de ce groupe, nous remercions plus particulièrement Frédéric Ferraty et Philippe Vieu pour nous avoir fourni les données sur lesquelles nous avons travaillé et plus généralement pour nous avoir donné accès à leurs

2. http://www.lsp.ups-tlse.fr/Fp/Ferraty/staph.html

travaux les plus récents. Nous remercions aussi Louis Ferré pour de fructueuses discussions sur nos travaux, en particulier concernant l'approche par projection. Nous remercions enfin les cinq rapporteurs anonymes dont les remarques et conseils ont contribué à améliorer le présent article.

6. Bibliographie

[ABR 03] Abraham C., Cornillon P.-A., Matzner-Lober E., Molinari N., Unsupervised Curve Clustering using B-Splines, *Scandinavian Journal of Statistics*, vol. 30, n^o 3, 2003, p. 581–595.

[BAR 93] Barron A. R., Universal Approximation Bounds for Superpositions of a Sigmoidal Function, *IEEE Trans. Information Theory*, vol. 39, n^o 3, 1993, p. 930-945,

[BAR 97] Bardos M., Zhu W. H., Comparaison de l'analyse disciminante linéaire et des réseaux de neurones – Applications à la détection de défaillance d'entreprises, *Revue de Statistique Appliquée*, vol. XLV, n^o 4, 1997, p. 65–92.

[BES 00] Besse P., Cardot H., Stephenson D., Autoregressive forecasting of some functional climatic variations, *Scandinavian Journal of Statistics*, vol. 4, 2000, p. 673-688.

[BES 03] Besse P., Cardot H., *Analyse des données (édité par Gérard Govaert)*, Chapitre 6 : Modélisation statistique de données fonctionnelles, p. 167–198, Hermès/Lavoisier, 2003.

[BIS 93] Bishop C., Curvature-driven smoothing : a learning algorithm for feedforward networks, *IEEE Trans. on Neural Networks*, vol. 4, n^o 5, 1993, p. 882–884.

[BIS 95] Bishop C., *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[BOS 00] Bosq D., *Linear Processes in Function Spaces : Theory and Applications*, vol. 149 de *Lecture Notes in Statistics*, Springer Verlag, 2000.

[BRE 84] Breiman L., Friedman J. H., Olshen R. A., Stone C. J., *Classification and Regression Trees*, Wadsworth, 1984.

[CAZ 75] Cazes P., Protection de la régression par utilisation de contraintes linéaires et non linéaires, *Revue de Statistique Appliquée*, vol. XXIII, n^o 3, 1975, p. 37–57.

[CIA 00] Ciarlet P. G., *Introduction à l'analyse numérique matricielle et à l'optimisation*, Dunod, Février 2000.

[CON 02] Conan-Guez B., Modélisation supervisée de données fonctionnelles par perceptron multicouches, Thèse de doctorat, Paris IX Dauphine, Décembre 2002.

[DAU 01] Dauxois J., Ferré L., Yao A.-F., Un modèle semi-paramétrique pour variables aléatoires hilbertiennes, *C. R. Acad. Sci. Paris*, vol. 333, 2001, p. 947–952.

[FER 01] Ferraty F., Vieu P., Statistique Fonctionnelle : Modèles de régression pour variables aléatoires uni, multi et infiniment dimensionnées, rapport n^{o}LSP-2001-03, 2001, Laboratoire de Statistique et Probabilités, Université Paul Sabatier, Toulouse, France.

[FER 02a] Ferraty F., Goia A., Vieu P., Régression non-paramétrique pour des variables aléatoires fonctionnelles mélangeantes, *C. R. Acad. Sci. Paris*, vol. 334, 2002, p. 217–220, Série I.

[FER 02b] Ferraty F., Vieu P., The Functional Nonparametric Model and Application to Spectrometric Data, *Computational Statistics*, vol. 17, n^o 4, 2002.

[FER 03a] Ferraty F., Vieu P., Curves Discriminations : a Nonparametric Functional Approach, *Computational Statistics and Data Analysis*, vol. 44, n^o 1–2, 2003, p. 161–173.

[FER 03b] Ferré L., Yao A.-F., Functional sliced inverse regression analysis, *Statistics*, vol. 37, n^o 6, 2003, p. 475–488.

[HAL 01] Hall P., Poskitt D., Presnell B., A functional data-analytic approach to signal discrimination, *Technometrics*, vol. 43, n^o 1, 2001, p. 1–9.

[HAS 94] Hastie T., Buja A., Tibshirani R., Flexible Discriminant Analysis by Optimal Scoring, *J. Amer. Statist. Assoc.*, vol. 89, 1994, p. 1255–1270.

[HAS 95] Hastie T., Buja A., Tibshirani R., Penalized Discriminant Analysis, *Annals of Statistics*, vol. 23, 1995, p. 73–102.

[HOE 70a] Hoerl A. E., Kennard R. W., Ridge regression : application to non orthogonal problems, *Technometrics*, vol. 12, n^o 2, 1970, p. 69-82.

[HOE 70b] Hoerl A. E., Kennard R. W., Ridge regression : Biased estimation for nonorthogonal problems, *Technometrics*, vol. 12, n^o 1, 1970, p. 55-67.

[HOR 93] Hornik K., Some new results on neural network approximation, *Neural Networks*, vol. 6, n^o 8, 1993, p. 1069-1072.

[JAM 00] James G. M., Hastie T. J., Sugar C. A., Principal component models for sparse functional data, *Biometrika*, vol. 87, n^o 3, 2000, p. 587–602.

[LES 93] Leshno M., Lin V. Y., Pinkus A., Schocken S., Multilayer Feedforward Networks With a Nonpolynomial Activation Function Can Approximate Any Function, *Neural Networks*, vol. 6, n^o 6, 1993, p. 861–867.

[MAR 89] Martens H., Naes T., *Multivariate Calibration*, John Wiley, New York, 1989.

[PRE 92] Press W. H., Teukolsky S. A., Vetterling W. T., Flannery B. P., *Numerical Recipes in C*, Cambridge University Press, second édition, 1992,

[RAM 97] Ramsay J., Silverman B., *Functional Data Analysis*, Springer Series in Statistics, Springer Verlag, June 1997.

[ROS 05] Rossi F., Conan-Guez B., Functional Multi-Layer Perceptron : a Nonlinear Tool for Functional Data Analysis, *Neural Networks*, vol. 18, n^o 1, 2005, p. 45–60.

[SAN 96a] Sandberg I. W., Notes on Weighted Norms and Network Approximation of Functionals, *IEEE Transactions on Circuits and Systems–I : Fundamental Theory and Applications*, vol. 43, n^o 7, 1996, p. 600–601.

[SAN 96b] Sandberg I. W., Xu L., Network approximation of input-output maps and functionals, *Circuits Systems Signal Processing*, vol. 15, n^o 6, 1996, p. 711–725.

[STI 99] Stinchcombe M. B., Neural network approximation of continuous functionals and continuous functions on compactifications, *Neural Networks*, vol. 12, n^o 3, 1999, p. 467-477.

[WHI 89] White H., Learning in Artificial Neural Networks : A Statistical Perspective, *Neural Computation*, vol. 1, n^o 4, 1989, p. 425-464,

[WHI 90] White H., Connectionist Nonparametric Regression : Mutilayer Feedforward Networks Can Learn Arbitrary Mappings, *Neural Networks*, vol. 3, 1990, p. 535-549,

[ZHU 97] Zhu W. H., Guicheney C., Berdagué J.-L., Jousset J., Application des réseaux perceptrons multicouches au contrôle de la qualité des aliments par analyse sensorielle – Comparaison des résultats avec différentes méthodes d'analyse discriminante, *Revue de Statistique Appliquée*, vol. XLV, n^o 2, 1997, p. 39–57.